Clustering with Expert Opinion

Xiaojin Zhu
LTI/CALD
Carnegie Mellon University
zhuxj@cs.cmu.edu

Tianjiao Chu
Philosophy/CALD
Carnegie Mellon University
tchu+@andrew.cmu.edu

Jie Zhu
LTI
Carnegie Mellon University
jzhu@cs.cmu.edu

February 15, 2001

Abstract

Traditional clustering algorithms cannot incorporate domain expert's opinion into the process of clustering. Since one major goal of clustering is for the experts to gain insight into a dataset, it makes sense to allow experts to have some degree of control over the cluster results. We propose a mechanism for the experts to request a small number of clustering constraints that they wish to be true. We devise an algorithm which takes into account the expert constraints and guides the clustering towards them, so that the final clustering satisfies the maximal number of expert constraints while is still reasonably good in traditional measures. We plan to work on the theory and implementation of this idea, and test it on both synthetic and real world datasets.

1 Introduction

Traditional clustering algorithms take the dataset, number of clusters, a distance function, and a set of distribution families (e.g. Gaussian) as inputs (some of which are optional). Then the algorithms start and finally output a clustering without human intervention. However in many cases one of the major purposes of clustering is for domain experts (e.g. astrophysicists, biologists, etc.) to gain insight into the datasets. Therefore it makes sense for the experts to be able to control the clustering in certain ways. Imagine a domain expert, besides giving the dataset to the computer, also specifies some relationships/constraints between a few data points in the dataset. Note "a few" is the key word: experts are busy, or the dataset is too large to be read by human. The expert may say the following:

- "data point A looks like it should be in the same cluster as datapoint B"
- "I'm pretty sure C and D are in different clusters"
- "I think E is much closer/similar to F than to G"

The clustering problem now becomes one of finding a good clustering that also mostly satisfies the constraints by human expert.

In its simplest form, this idea can be applied to attribute weighting. Let's assume data points are represented as k dimensional vectors of continuous attributes, e.g. $A = (a_1, a_2, \ldots, a_k)$ Under this condition, most literature assumes a given distance function between two data points, usually the Euclidean distance: $dist(A, B) = \sqrt{\sum_{i=1}^{k}(a_i - b_i)^2}$ This assumes that the dataset has been properly prepared by the domain expert, so that each attribute has appropriate scale and the distance makes sense. However imaginably this can go wrong. For instance consider a health care dataset containing two attributes 'weight' and 'height' in kg and cm respectively:

person1 (50, 160) person2 (55, 165) person3 (51, 190)

Common sense tells us that person 1 and person 2 are more or less normal, while person 3 is abnormal. The Euclidean distance coincides with our intuition since e.g. $dist(person1, person2) = 7.07 < 30.02 = dist(person1, person3)$. But if the height is in m (meter):

person1 (50, 1.60) person2 (55, 1.65) person3 (51, 1.90)

Now $dist(person1, person2) = 5.00 > 1.04 = dist(person1, person3)$, i.e. Euclidean distance no longer matches our intuition. In this case different unit on an attribute results in different distance measures, and may give rise to unexpected clustering to a health care expert. Note normalizing individual attributes or performing principal component analysis on the dataset can't solve this problem. We believe this situation will often occur, since it is very hard for the domain expert to choose appropriate attribute scales. Nevertheless, it should be relatively easy for the expert to give a small set of constraints as above, e.g. "person 3 is not similar to person 1, while person 2 is". The hope is that we can find a weight vector $W = (w_1, \ldots, w_k)$ that stretches the attributes:

$$dist(A, B) = \sqrt{\sum_{i=1}^{k} w_i^2 (a_i - b_i)^2}$$

that gives rise to a clustering which (mostly) satisfies the expert opinion. The goal of the proposed project is to find a method to incorporate expert opinion into clustering process to produce better clusters.

2 Survey

2.1 Old fashioned Clustering

2.1.1 Statistical Clustering: Parametrical or Nonparametric

One of the classical references of clustering algorithm is Hartigan [7]. Chapter 5 of this book introduces the general method of model based clustering algorithm. This approach assumes that we know the number of clusters, say, k. The d-dimensional data points are assumed to be generated from a distribution with density function $f(x_1, \cdots, x_d)$ such that:

$f(x_1, \cdots, x_d) = \sum_{i=1}^{k} p_i f_i(x_1, \cdots, x_d | \vec{\theta}_i)$

where $p_i > 0$, $\sum_i p_i = 1$, and f_i is the density function of the ith distribution, with parameter vector $\vec{\theta}_i$.

An iterative algorithm is proposed to find the values of p_i's and $\vec{\theta}_i$'s. As we now know, it is a version of EM algorithm. (This book was published two years before Demptster et al. [5]). In particular, the author talks about the case where the f_i's are assumed to be the density functions of multivariate normal distributions. To reduce the number of parameters to be estimated, the author also presents several ways to simplify the general algorithm. Basically, the author imposes more constraints on the density functions, e.g. they share the same covariance matrices, the covariance matrices are diagonal, etc.

Fraley and Raftery [6] presents a modern approach to the clustering problem based on parametric models. In particular, they tackle the problem of number of clusters. The classical approach assumes that we already know the number of clusters, which, in practice is often not the case. Another problem of the classical approach is that, although we could impose various constraints on the density functions to get different parametric models, in general, we do not know which model is the best one. To deal with these problems, Fraley and Raftery propose to run the EM algorithm on different number of clusters and different kind of parametric models. (To speed the rate of convergence, the input to the EM algorithm is preprocessed by the agglomerative hierarchical clustering method). Then, the BIC score function is used to compare the fitted models. They claim that this approach, although computationally expensive, gives better performance than the classical method.

The model based clustering algorithms assume that all the clusters have some regular shapes. For example, all the mixture of normal models based clustering algorithms assume that all the clusters should be either spherical or ellipsoidal. In practice, there are many cases where this assumption is not even approximately true, e.g., the problem of pattern recognition. In these cases, the nonparametric approach to clustering is more appropriate.

Cuevas et al. [3] presents a clustering algorithm based on nonparametric density estimation. The basic idea of this approach is that, each cluster is assumed to be a connected component of the set $S = \{\vec{x} : f(\vec{x}) > c\}$, where f is the density of the underlying distribution, and c is some positive number. To find all the components in S, the author suggests the following algorithm: First, estimate the density $\hat{f}$ from the sample using some density estimation method, e.g., kernel density estimation. Choose a positive c based on some criterion, e.g., $P(\{\vec{x} : f(\vec{x}) \leq c\}) = a$, where a is some positive number between 0 and 1. (The larger the a is, the weaker the cluster structure is.) Set a radius $\epsilon > 0$. (The authors give some suggestion regarding how to choose ϵ.) Then include 2 points $\vec{x}_i$ and $\vec{x}_j$ in the same cluster if and only if $|\vec{x}_i - \vec{x}_j| < 2\epsilon$, $\hat{f}(\vec{x}_i) > c$, and $\hat{f}(\vec{x}_j) > c$. Bootstrap method can be used to estimate the variability of the result of the clustering algorithm.

2.1.2 Agglomerative and Divisive Clustering

For data points that are characterized by distances between them instead of coordinates, clustering is generally more difficult. The most intuitive and popular approach is hierarchical clustering: at each level a single point or group of points is joined with another (agglomerative methods), or a group of points is divided (divisive methods). Agglomerative methods are usually preferred since they require less computation. In [8], Hoffman and Buchmann applied the deterministic annealing technique to pairwise clustering. Their method uses simulated annealing to stochastically optimizing a cost function, which is derived according to maximum entropy inference. It is an influential work in pairwise clustering.

Tishby et al. recently introduced another method, which they call information bottleneck method [13] [12]. In their approach, the clustering problem is formalized as finding a short representation Y for one set of points X that preserves the maximum information about another set Z (Z can be a set of given categories). This yields a variational problem of maximizing the mutual information I(Y,Z) between Y and Z, subject to a constrain on the mutual information I(X,Y) between X and Y. In such a formulation, I(Y,Z) represents the information about Z encoded in Y, and I(X,Y) shows how compact the representation Y of X is. They implemented this idea as an iterative re-estimation method. Although their method is computationally heavy, their idea provides a rich framework for discussing a variety of problems in learning.

2.2 Large Dataset Clustering

The BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) system, which is introduced in [14] by Zhang et al., dismisses EM algorithm because of the large amount of data accesses required by EM. It incrementally and dynamically clusters incoming multi-dimensional metric data points to produce the best quality clustering with the available computing time and memory resources. By doing incremental clustering based on an in-memory, height-balanced and highly-occupied clustering feature tree, it minimizes the number of I/O operations and memory requirements. Each clustering decision is made without scanning all data points or all currently existing clusters, and is thus faster than global clustering methods. It's popular in the database community because of its ability to deal with very large data sets with relatively limited resources.

The problem of BIRCH is that it is unable to identify second-moment features of clusters, such as non-axis-aligned spread. In [11], Moore came up with an EM-based mixture model clustering algorithm using the multiresolution kd-trees of [10] that greatly reduced the computation and I/O cost of traditional EM-based methods. This algorithm constructs an mrkd-tree top-down, identifying the bounding box of the current node, and splitting in the center of the widest dimension. By computing the minimum and maximum ownership that any point inside the node could have, many prunes can be made, and thus reduce the search space at every iteration of EM. With this saving on I/O operations and computations, we can use our good-old EM-based mixture model clustering to deal with larger data sets.

In [9] Pelleg and Moore extended the popular K-means with an estimation of the number of clusters. The resulting algorithm (named X-means) adds an "improve-structure" operation to the conventional K-means algorithm: in each iteration, after the K-means converge, find out if and where new centroids should appear. This new-centroids searching process is as follows: split every cluster centroids, do K-means in every cluster with the new centroids; then accept those new centroids that pass a Bayesian Information Criterion based test. The whole process, including the K-means step and the improve-structure step, is accelerated by operations similar to those used in [11]. This algorithm shows good performance in median dimensionality (4 to 7) problems.

2.3 New Direction in Clustering: Put Human in the Loop

As we stated earlier, one major goal of clustering is for the domain experts to gain insight into the dataset. All clustering methods discussed above make assumptions, e.g. the distance function, the distribution, etc., which may not hold in practice. As a result, the single 'best' clustering may not truly reflect the nature of the dataset.

Metaclustering [1] is a new technique that addresses this problem. Instead of find the single 'best' clustering, metaclustering generates a large number of different but reasonable clusterings, and let the users (domain experts) pick a satisfactory one. To help users examine the vast variety of clusterings, metaclustering clusters clusterings according to some predefined similarity measure on clusters (hence the name) before presenting them to the users. As a result, similar clusterings are grouped together, and the users only need to examine the typical ones of each group. In this paper, multiple clusterings were generated from different samples of the dataset by sampling with replacement (the so called bagging). They tried several different similarity measures between clusterings, the one based on hop distance seemed good. The basic clustering method was agglomerative clustering. They tried metaclustering on a synthetic dataset and the protein dataset with promising results. The main advantage of metaclustering is that users can actively choose the clustering they want with relatively small effort. The disadvantage is efficiency, since it has to generate a large number of different clustering, even though most of which may not be favored by the users and are never looked at. Unlike our proposed method here, the domain experts has no control over the process before everything has settled down.

Cohn, Caruana and McCallum proposed "semi-supervised clustering" [2] which has almost exactly the same basic idea as our proposed work: let the domain experts express their knowledge in the form of a few constraints on some data points, and the clustering algorithm takes into account these constraints to find a clustering that are both good in itself and in satisfying the constraints. Unfortunately their work was unpublished and we independently come up with the same idea later. Still, our proposed work has some important differences: First, they only implemented constraints of the form "point A and B should not be in the same cluster", while we plan to support more kinds of constraints (see sections below); Second, they worked on text document clustering problem where only the distance between documents is known, but we assume explicit vector representation of data points; In addition they assume an iterative interaction between the clustering algorithm and the user, while we assume all constraints are given a priori. And finally they modify the distance function between documents, while we weight attributes.

Demiriz, Bennett and Embrechts [4] also worked on semi-supervised clustering problems. Their work is more close to a classification with unlabeled data setting. They assume the user can provide class labels to a few data points, which is equivalent to having a small training set. Then they perform clustering on the whole dataset, with the goal of finding a good clustering that are 'pure', i.e. datapoints within one cluster should have the same class label. The clustering can then be used to induce the classification on unlabeled data. They used genetic algorithms to find such clusterings. However the assumption that user can provide a few class labels is qualitatively different from our assumption on expert opinions. By class labels they implicitly assume the user knows how many classes there are, and which sample belongs to which class. This is a much stronger assumption. In contrast, we don't have the notion of classes. The user only need to specify whether two data points are similar, should be in the same cluster, or should not be in the same cluster. There is no need to assign class to data points. Moreover, the relationship between classes and clusters could be complex: a class can contain several clusters, or a cluster can have data points from several classes. We believe our scheme would be much easier for the user.

3 Technical Approach

To simplify the problem, we make the following assumptions:

- Data points are represented in vector form $A = (a_1, a_2, \dots, a_k)$, each element of the vector is a continuous valued attribute.
- The number of clusters is predetermined and fixed.
- We are given a working clustering algorithm (e.g. EM with mixture of Gaussian, or agglomerative clustering algorithm) which we will use as a black box.

Prior to clustering, the expert examines the dataset and gives a set of constraints $C = \{c_1, \dots, c_m\}$. Each constraint c_i can have one of the following form:

1. Data points A and B shall be in the same cluster.

2. A and B shall be in different clusters.

3. A, B are closer than A, C.

Our objective is to find a weight vector $W = (w_1, \ldots, w_k)$ that stretches/compresses attribute values. The weight will be applied to all data points $A' = A \cdot W = (w_1 a_1, w_2 a_2, \ldots, w_k a_k)$ so that the weighted dataset, when given to the black box clustering algorithm, results in a clustering that satisfies the maximal number of constraints.

We propose to use an iterative process and heuristic functions to find such a weight vector. We start from a unit weight vector $W^0 = (1, 1, \ldots, 1)$ and cluster the dataset. For each expert constraint that is violated by the clustering, we adjust the weight vector heuristically as follows:

1. If the expert expects A, B to be in the same cluster but they are not, we would like to move A, B closer with respect to other points. Note we don't want to simply reduce the magnitude of W since this more or less shrinks the whole space evenly, and the relative distance between A and B essentially doesn't change. We will experiment with several heuristics, including:
   - Find the dimension (attribute) j such that A and B differs most: $arg\max_j |a_j - b_j|$, reduce w_j the weight in this dimension.
   - Pick a dimension j randomly, with probability proportional to $|a_j - b_j|$. Reduce w_j.
   - Pick a dimension j such that reducing w_j has minimal adverse effect on other expert constraints. For example, if the expert also want C and D to be in the same cluster but they are not, we hope our reduction on w_j won't increase the distance between C, D.

2. If the expert wants A, B to be in different clusters but they are in the same cluster now, we increase the distance between A and B by heuristics similarly.

3. If the expert wants A, B to be closer than A, C but it is not, we update $w_i, i = 1, \ldots, k$ as

$$w_i^{t+1} = w_i^t - \eta \frac{\partial}{\partial w_i}(dist(A, B) - dist(A, C))$$

After adjusting weight vector for every violated constraint, we apply the new weight vector to the dataset and cluster it again. We hope the new weight leads to smaller number of violations. The weight update process is then repeated until converge or reaches predefined number of iterations. The procedure is shown in Figure 1.

4 Work Plan

Our project will be a joint work in the genuine sense. We have spent, and will spend, a lot of time to discuss about ideas coming from various areas familiar to one or more of us, examine the relevance of

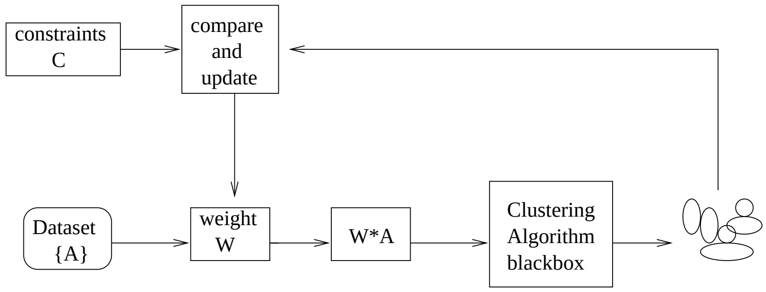

Figure 1: Flow chart of the proposed method

these ideas to our project, and explore how these ideas could be incorporated into our approach. It is not always possible, in this work plan, to state explicitly which one of us will do which part of work. But we will try to specify how to divide some work among us when it makes sense.

Our project will be implemented in three stages:

- First stage: Feb. 1 — Feb. 15 Search for the literatures relative to our goal. Discuss possible approaches. Write the proposal.

  Jerry Zhu: Search for the literature about use of user input in clustering. Write the survey for this part of literature.

  Tianjiao Chu: Search for the statistical approaches to clustering algorithm. Write the survey for this part of literature.

  Jie Zhu: Search for the literature in large database clustering. Write the survey for this part of literature.

- Second stage: Feb. 19 — April 3 Implement the algorithms. Write the progress report.

  Jie Zhu: Collect data, implement the synthetic data generation algorithm.

  Tianjiao Chu: Implement the clustering algorithm.

  Jerry Zhu: Implement the algorithm to evaluate the clustering results.

- Third stage: April 4 — May 3 Test the algorithm both with synthetic data and real data. Evaluate the results. Discuss their implications and possible future works. Write the final report.

  Jie Zhu: Generate the data.

  Jerry Zhu: Create the user inputs.

  Tianjiao Chu: Run the algorithm.

5 Summary

We proposed a method to incorporate expert opinions into the process of clustering. The proposed research, if successful, will allow experts to specify domain knowledge with little effort and get satisfactory clustering results. We believe it will be a valuable extension to standard clustering algorithms, and domain experts will want to keep it in their collection of datamining tools.

References

[1] Rich Caruana, Pedro Artigas, Anna Goldenberg, and Anton Likhodedov. Meta clustering. submitted for publication, ICML2001.

[2] David Cohn, Rich Caruana, and Andrew McCallum. Semi-supervised clustering with user feedback, 2000. draft.

[3] A. Cuevas, M. Febrero, and F. Fraiman. Estimating the number of clusters. *The Canadian Journal of Statistics*, 2000.

[4] A. Demiriz, K. Bennett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. In *ANNIE*, 1999.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(B):1–38, 1977.

[6] C. Fraley and A. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *Computer Journal*, 41:578–588, 1998.

[7] J. Hartigan. *Clustering Algorithms*. John Wiley & Sons, New York, 1975.

[8] T. Hoffmann and J. Buchmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. on PAMI*, 19(1):1–14, 1997.

[9] A. Moore and D. Pelleg. X-means: Extending K-means with efficient estimation of the number of clusters. In *Proceedings of the International Conference on Machine Learning*, 2000.

[10] A. Moore, J. Schneider, and K. Deng. Efficient locally weighted polynomial regression predictions. In *Proceedings of the 1997 International Machine Learning Conference*, 1997.

[11] Andrew Moore. Very fast mixture-model-based clustering using multiresolution kd-trees. In *Advances in Neural Information Processing Systems*, pages 543–549, 1999.

[12] N. Slonim and N. Tishby. Agglomerative information bottleneck. In *Proceedings of Neural Information Processing Systems*, 1999.

[13] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[14] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large database. In *Proceedings of ACM SIGMOD*, pages 103–114, 1995.